Structural Jacobian Accumulation with Unit Edges

Andrew Lyons
Computation Institute, University of Chicago
email: lyonsam@gmail.com

Optimal Jacobian accumulation

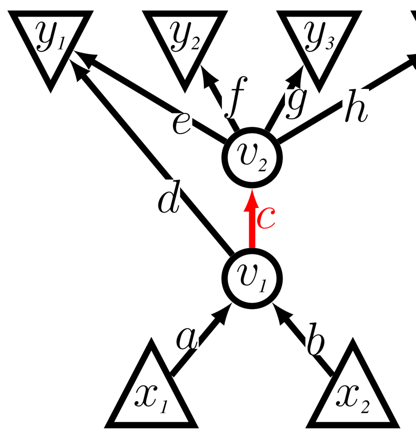
Given the graph G for vector function $\mathbf{y} = F(\mathbf{x})$, $F : \mathbb{R}^n \to \mathbb{R}^m$, *Baur's formula* yields the entries of the Jacobian as

$$J_{j,i} \equiv \frac{\partial y_j}{\partial x_i} = \sum_{P \in \mathcal{P}_{[x_i \to y_j]}} \prod_{e \in P} c_e \ ,$$

where $\mathcal{P}_{[x_i \to y_j]}$ denotes the set of all paths from x_i to y_j in G.

Example 1. $y_1 = (x_1 * x_2)/(x_1 * x_2 + 10)$, $y_2 = \sqrt{x_1 * x_2 + 10}$, $y_3 = \sin(x_1 * x_2 + 10)$, $y_4 = \cos(x_1 * x_2 + 10)$. $J_{1,1} = ad + ace$, $J_{1,2} = bd + bce$, $J_{2,1} = acf$, $J_{2,2} = bcf$, $J_{3,1} = acg$, $J_{3,2} = bcg$, $J_{4,1} = ach$, $J_{4,2} = bch$.

$$\begin{aligned} v_1 &= x_1 * x_2 \\ v_2 &= v_1 + 10 \\ y_1 &= v_1 / v_2 \\ y_2 &= \sqrt{v_2} \\ y_3 &= \sin(v_2) \\ y_4 &= \cos(v_2) \end{aligned}$$



$$\begin{aligned} &t_1 = c * e; \quad t_2 = d + t_1; \\ &J_{1,1} = a * t_2; \ J_{1,2} = b * t_2; \\ &t_3 = a * c; \quad t_4 = b * c; \\ &J_{2,1} = t_3 * f; \ J_{2,2} = t_4 * f; \\ &J_{3,1} = t_3 * g; \ J_{3,2} = t_4 * g; \\ &J_{4,1} = t_3 * h; \ J_{4,2} = t_4 * h; \end{aligned}$$

Problem 1. STRUCTURAL OPTIMAL JACOBIAN ACCUMULATION (SOJA)
Instance: *Dag $G = (V, E)$, where each $e \in E$ is labeled with some c_e such that all c_e are unique real variables that are algebraically independent, positive integer K.*
Question: *Is there a straight-line program using operations in $\{+, *\}$ of length K or less that computes every entry in J such that every operand is either some c_e or the result of a previous operation?*

Problem 2. OPTIMAL JACOBIAN ACCUMULATION (OJA)
Instance: *Dag $G = (V, E)$, where each $e \in E$ is labeled with some c_e that represents a real variable, positive integer K.*
Question: *Is there a straight-line program using operations in $\{+, *\}$ of length K or less that computes every entry in J such that every operand is either some c_e or the result of a previous operation?*

Theorem 1 (Naumann 2008)**.** OJA *is* **NP**-*hard.*

Corollary 1 (Naumann 2008)**.** OJA* *is* **NP**-*hard.*

Problem 3. STRUCTURAL OJA WITH UNIT EDGES (SOJA$_1$)
Instance: *Dag $G = (V, E)$, where each $e \in E$ is labeled with either a unique real variable c_e or a positive or negative unit label $+/-1$ such that all c_e are algebraically independent, positive integer K.*
Question: *Is there a straight-line program using operations in $\{+, *\}$ of length K or less that computes every entry in J such that every operand is either the label on some $e \in E$ or the result of a previous operation?*

The complexity of SOJA$_1$ and SOJA$_1^+$

An instance of ENSEMBLE COMPUTATION consists of a finite set S, a collection $C = \{C_1, C_2, \ldots, C_r\}$ of distinct subsets of S, and a positive integer K. It is **NP**-hard to decide whether the sets in C can be constructed from the elements of S using K or fewer disjoint union operations.

Lemma 1. SOJA$_1^+$ *is* **NP**-*hard.*

Proof. Reduction from ENSEMBLE COMPUTATION. □

Theorem 2. SOJA$_1$ *is* **NP**-*hard.*

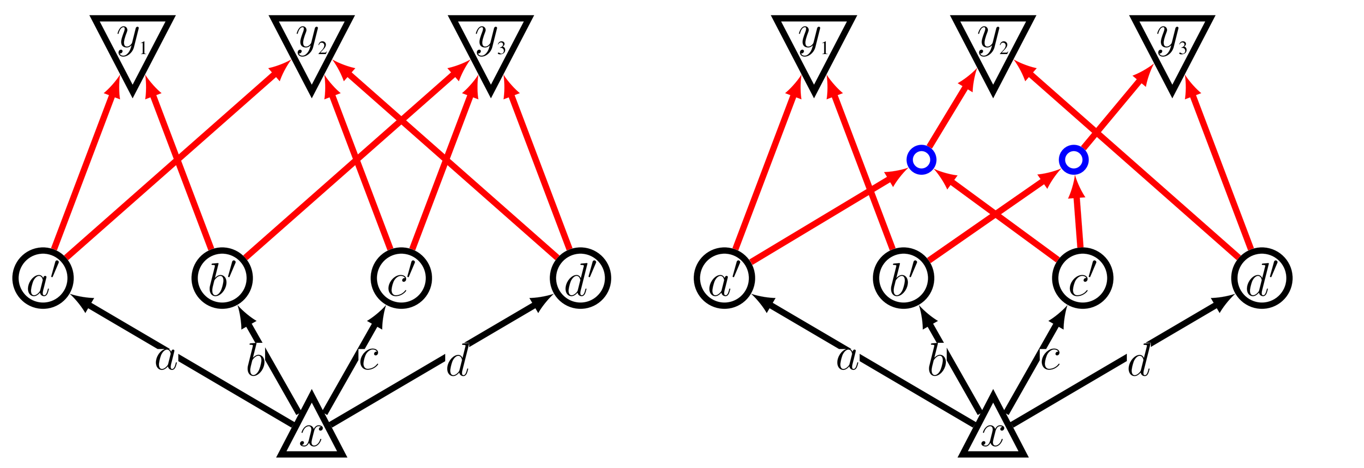
Proof. The instances of SOJA$_1$ that we construct for the above lemma involve additions exclusively. □

Corollary 2. SOJA$_1$ *and* SOJA$_1^+$ *remain* **NP**-*hard under each of the following restrictions.*

(i) G represents a tangent or gradient and all paths in G have length ≤ 2.

(ii) G represents a tangent or gradient, all vertices in G have indegree ≤ 2, and all paths in G have length ≤ 3.

Example 2. The following graphs correspond to the ENSEMBLE COMPUTATION instance $S = \{a, b, c, d\}$, $C = \{\{a, b\}, \{a, c, d\}, \{b, c, d\}\}$, $K = 4$. The answer to this instance is YES, as all sets in C are yielded by the following collection of four union operations. $T = \{c\} \cup \{d\}$; $C_1 = \{a\} \cup \{b\}$; $C_2 = \{a\} \cup T$; $C_3 = \{b\} \cup T$.



The complexity of SOJA$_1^*$

Problem 4. PARTITION INTO COMPLETE BIPARTITE SUBGRAPHS
Instance: *Bipartite graph $G = (A, B, E)$, positive integer K.*
Question: *Can the edges of G be partitioned into $k \leq K$ disjoint sets $C_1, C_2, \ldots, C_k$ such that each C_i is a complete bipartite graph?*

Theorem 3 (Gonzalez and JáJá 1980)**.** PARTITION INTO COMPLETE BIPARTITE SUBGRAPHS *is* **NP**-*complete.*

Theorem 4. SOJA$_1^*$ *is* **NP**-*hard.*

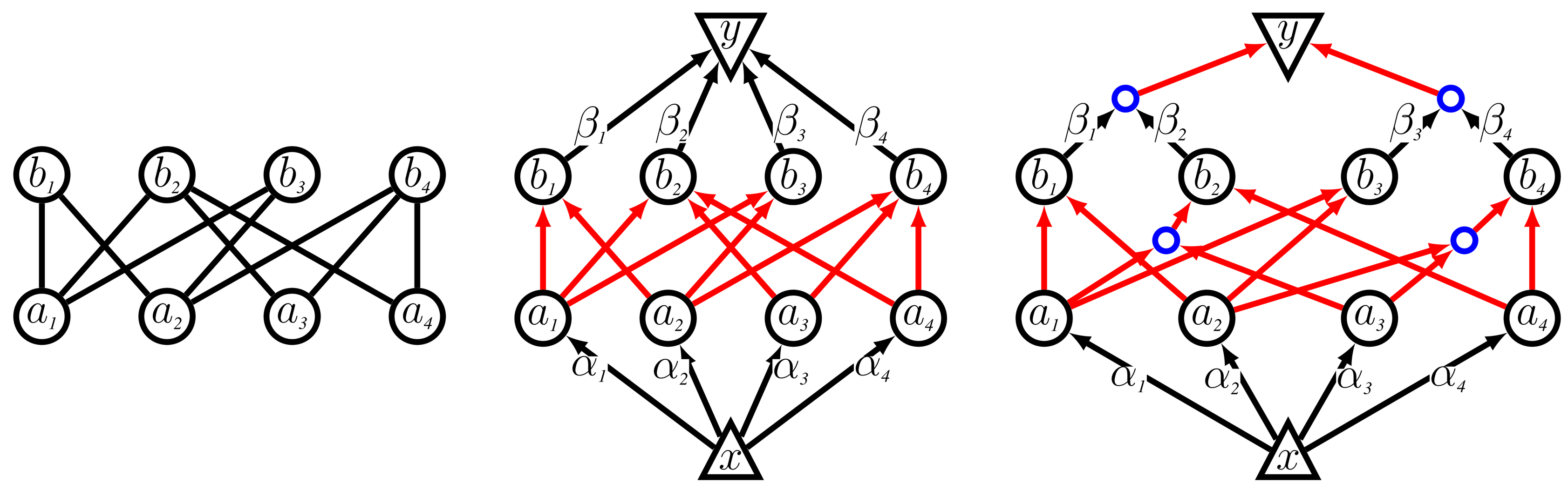
Proof. Reduction from PARTITION INTO COMPLETE BIPARTITE SUBGRAPHS. □

Corollary 3. SOJA$_1^*$ *remains* **NP**-*hard under each of the following restrictions.*

(i) G represents a scalar Jacobian and all paths in G have length ≤ 3.

(ii) G represents a scalar Jacobian and all vertices in G have indegree ≤ 2.

Example 3. *An instance of* PARTITION INTO COMPLETE BIPARTITE SUBGRAPHS *and the corresponding instances of* SOJA$_1$*. Accumulating J as $(\alpha_1 + \alpha_2)(\beta_1 + \beta_3) + (\alpha_1 + \alpha_3 + \alpha_4)\beta_2 + (\alpha_2 + \alpha_3 + \alpha_4)\beta_4$ carries a cost of three multiplications, which is the minimum for this example, and corresponds to the optimal biclique partition.*

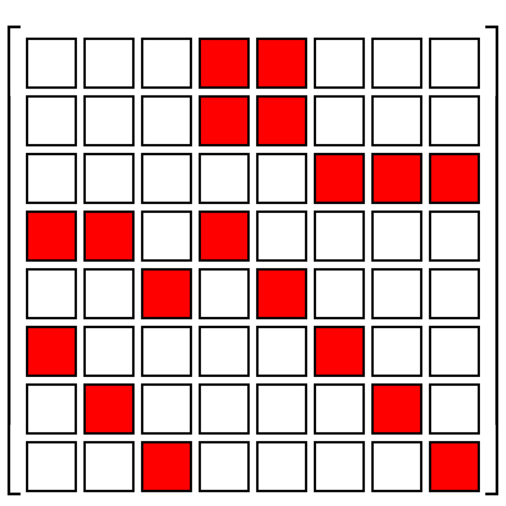


The utility of subtraction

Gonzalez and JáJá (1980) considered the optimal evaluation of bilinear forms $B = \boldsymbol{\alpha}^T R \boldsymbol{\beta}$, where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_p)^T$, $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_q)^T$, and R is a $p \times q$ matrix whose elements are all in $\{0, 1\}$. As the following example demonstrates, any bilinear form $B = \boldsymbol{\alpha}^T R \boldsymbol{\beta}$ can be expressed as an instance of SOJA$_1$ such that accumulating the (scalar) Jacobian J yields the value of B.

Example 4. *The following example is due to Gonzalez and JáJá. It is shown that B can be computed with only six multiplications using operations in $\{+, -, *\}$, whereas seven multiplications are required when using operations in $\{+, *\}$.*

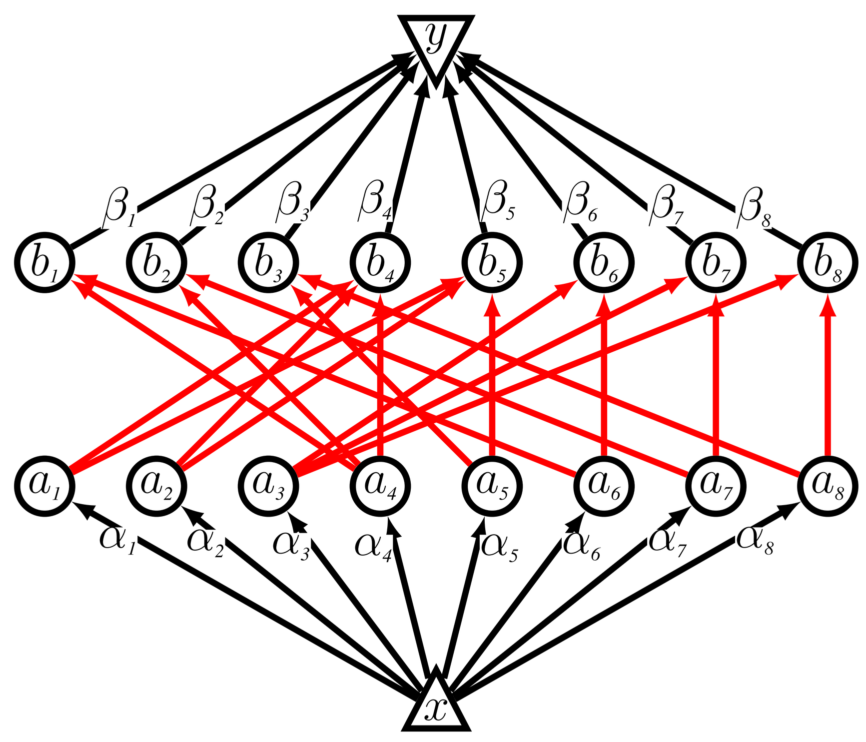
$B = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6\ \alpha_7\ \alpha_8]$



$[\beta_1\ \beta_2\ \beta_3\ \beta_4\ \beta_5\ \beta_6\ \beta_7\ \beta_8]^T$

$$\begin{aligned} &= \alpha_1\beta_4 + \alpha_1\beta_5 + \alpha_2\beta_4 + \alpha_2\beta_5 + \alpha_3\beta_6 + \alpha_3\beta_7 + \alpha_3\beta_8 + \alpha_4\beta_1 + \alpha_4\beta_2 \\ &\quad + \alpha_4\beta_4 + \alpha_5\beta_3 + \alpha_5\beta_5 + \alpha_6\beta_1 + \alpha_6\beta_6 + \alpha_7\beta_2 + \alpha_7\beta_7 + \alpha_8\beta_3 + \alpha_8\beta_8 \end{aligned}$$

$$\begin{aligned} &= (\alpha_1 + \alpha_2 + \alpha_3)(\beta_1 + \beta_2 + \beta_4) + (\alpha_3 + \alpha_6)(\beta_1 + \beta_6) + (\alpha_1 + \alpha_2 + \alpha_5)(\beta_3 + \beta_5) \\ &\quad + (\alpha_3 + \alpha_7)(\beta_2 + \beta_7) + (\alpha_3 + \alpha_8)(\beta_3 + \beta_8) - (\alpha_1 + \alpha_2 + \alpha_3)(\beta_1 + \beta_2 + \beta_3) \ . \end{aligned}$$



$$\begin{aligned} J = &(\alpha_1 + \alpha_2 + \alpha_3)(\beta_1 + \beta_2 + \beta_4) + (\alpha_3 + \alpha_6)(\beta_1 + \beta_6) + (\alpha_1 + \alpha_2 + \alpha_5)(\beta_3 + \beta_5) \\ &+ (\alpha_3 + \alpha_7)(\beta_2 + \beta_7) + (\alpha_3 + \alpha_8)(\beta_3 + \beta_8) - (\alpha_1 + \alpha_2 + \alpha_3)(\beta_1 + \beta_2 + \beta_3) \ . \end{aligned}$$